

Extracting Actionable Insights from User Movie Reviews

Submitted to Dr. Santosh Singh Rathore

Submitted By :
BHAVANA MEKALA
2021BCS-041



TABLE OF CONTENTS

- INTRODUCTION
 - METHODOLOGY
 - APPROACH - 1
 - APPROACH – 2
 - COMPARISON
- 

Introduction

- Understanding customer sentiments and feedback is crucial for businesses to improve their products and services.
- Movie reviews on platforms like IMDB provide valuable insights into audience opinions.
- Automating the analysis of these reviews can save time and provide actionable insights, helping filmmakers and producers understand audience preferences and improve future productions.

Objective

- To develop a sentiment classification model which classifies the movie reviews into positive and negative sentiments.
- To apply topic modeling techniques to the subset of negative reviews in order to uncover recurring themes and concerns expressed by users.

Methodology

Data preprocessing

- Converting all text to lowercase to ensure uniformity. Removing URLs, digits, HTML tags, punctuation, and stop words to eliminate noise in the data. Reducing words to their base forms to consolidate variations and improve analysis quality. Eliminating common and frequent words that do not add significant meaning to the text, further refining the dataset.

Sentiment Analysis

- After preprocessing, we perform sentiment analysis to classify the reviews into positive and negative sentiments. This classification helps in understanding the overall user sentiment toward the films.

Topic Modelling

- Following sentiment analysis, we perform Topic modeling specifically on the negative reviews. This aims to identify common themes and topics, providing deeper insights into the issues highlighted by users. We evaluate its performance using the coherence score.

How VADER works?

VADER - Valence Aware Dictionary for sEntiment Reasoning

- It has a pre-defined list of words (lexicon), each associated with a sentiment score. Positive words (e.g. "happy") have positive scores, negative words (e.g. "sad") have negative scores, and neutral words have scores close to zero. When analyzing text, VADER looks up each word in this lexicon to get its sentiment score.
- Exclamation marks increase the intensity of the sentiment. Example, "great!!!" is more positive than just "great". Uppercase words are considered to have more emphasis. For example, "GREAT" is stronger than "great".
- It has rules to handle negation words like "not" and "never". For instance, "not good" would flip the sentiment of "good" from positive to negative. It looks at words around the negation to understand its effect.
- Words that amplify or dampen sentiment (like "very", "extremely", "slightly") are taken into account. Example, "very happy" would have a higher positive score than just "happy". VADER adjusts the sentiment score based on these modifiers.

Step-By-Step Process

- The text is split into individual words or tokens. Each token is then analyzed using the lexicon.
- VADER assigns a sentiment score to each word based on the lexicon.
- Scores are then adjusted based on punctuation, capitalization, negation, and degree modifiers.
- After scoring each word, VADER combines these scores to give an overall sentiment score for the entire text.
- The overall sentiment score is a combination of positive, negative, and neutral scores.
- VADER outputs the final sentiment as a compound score (a single number that indicates overall sentiment) and also provides individual positive, negative, and neutral scores.

Example

"I am VERY happy with this product!!!"

- Tokenization: ["I", "am", "VERY", "happy", "with", "this", "product", "!!!"]
- Lexicon Lookup:
 - "happy" has a positive score.
 - "VERY" is a degree modifier that amplifies "happy".
- Punctuation:
 - The exclamation marks "!!!" increase the intensity of "happy".
- Combining Scores:
 - The positive score for "happy" is increased by "VERY" and the exclamation marks.
- Final Sentiment:
 - The overall sentiment is strongly positive.

Issues with VADER

- VADER uses a fixed list of words to understand sentiment. If the text has words or phrases not on this list it might not understand the sentiment correctly.
- It looks at words individually or in small groups without understanding the bigger picture. This means it can miss the meaning of words that change based on context.
- It has rules to handle negation (like "not good"), but it can struggle with more complicated or subtle negations.
- It has a hard time recognizing sarcasm and irony, often leading to wrong sentiment scores.
- VADER is meant for general use and might not work well for specific fields like legal or technical documents.
- VADER can't learn from new data. It doesn't get better with more information or adapt to new language trends.

Alternative ML-Based Methods

- Support Vector Machines (SVM)
- Naïve Bayes
- Logistic Regression
- Random Forests

SVM

Advantages :

- Works well with small data
- Works well with a clear margin of separation

Drawbacks :

- SVMs can be slow to train especially with large datasets.
- Requires extensive preprocessing
- Less interpretability

Performance :

- Can outperform VADER when trained on a large and representative dataset, capturing nuances that rule-based systems miss.

Naïve Bayes

Advantages :

- Simple and fast
- Performs well with small and medium-sized datasets

Drawbacks :

- Assumes independence between features, which is rarely true in text data

Performance :

- Often performs competitively with VADER straightforward sentiment analysis tasks. May struggle with more complex sentiment nuances

Logistic Regression

Advantages :

- Simple and interpretable
- Effective for binary classification

Drawbacks :

- Requires good feature engineering
- Assumes Linearity which might not capture complex patterns

Performance :

- Can perform better than VADER, especially with good feature extraction. Struggles with very complex sentiment contexts

Random Forests

Advantages :

- Robust to overfitting with a large number of trees
- Can handle non-linear data
- Provides insights into which features are most important

Drawbacks :

- Computationally Intensive
- Less interpretable

Performance :

- Often performs VADER by capturing complex patterns in data, provided sufficient training data

Which is Better?

- Naïve Bayes is a solid choice if we need a balance between simplicity, speed and reasonable accuracy. Its particularly strong if you have a large dataset and want something interpretable
- SVM is a good option if we prioritize accuracy and have computational resources to handle the training process
- Logistic Regression is preferable if we have interpretability and a straightforward probabilistic model
- Random forests are ideal for achieving the highest accuracy, especially in cases where the relationship between features are complex, but at the cost of increased computational resources and interpretability

Approach-1

Sentiment Analysis by
Naive Bayes followed
by LDA for Topic Modeling

Bayesian Classifier

- Naive Bayes is a probabilistic classifier based on Bayes' theorem, which assumes independence between predictors. Bayes Theorem is a mathematical formula used to determine the conditional probability of events.
- Each feature in the text is considered independent of others. Features are words in the review. A word not seen in training data might make $P(\text{word/class}) = 0$. To handle this use Laplace smoothing to ensure no probability is zero.
- $P(\text{word/class}) = (\text{count of word in class} + 1) / (\text{total words in class} + V)$
- V is the unique word in the corpus.

Example

	Category	Documents
Training	-	Just plain boring
	-	Entirely predictable and lacks energy
	-	No surprises and very few laughs
	+	Very powerful
	+	The most fun film of the summer
Test	?	Predictable with no fun

- Prior from training :

$$P(-) = 3/5 \quad \text{and} \quad P(+) = 2/5$$

- Drop with

- Likelihoods from training :

$$P(\text{predictable} / -) = 1 + 1 / 14 + 20$$

$$P(\text{predictable} / +) = 0 + 1 / 9 + 20$$

$$P(\text{no} / -) = 1 + 1 / 14 + 20$$

$$P(\text{no} / +) = 0 + 1 / 9 + 20$$

$$P(\text{fun} / -) = 0 + 1 / 14 + 20$$

$$P(\text{fun} / +) = 1 + 1 / 9 + 20$$

- Scoring the test set :

$$P(-)P(s/-) = 3/5 * 2*2*1/34 * 34*34 = 6.1 * 10^{-5}$$

$$P(+)P(s/+) = 2/5 * 1*1*2/34 * 34*34 = 3.2 * 10^{-5}$$

Naïve Bayes Variants

- Multinomial NB (Bayesian Classifier) is used for discrete data, particularly word counts in text. Assumes features are the frequencies or counts of words. It is used to estimate the likelihood of seeing a specific set of word counts in a document. Widely applied for document classification and sentiment analysis.
- Complement NB is particularly suited to address imbalances in the class distribution. A variant of Multinomial NB, this model adjusts the calculation to complement the existing model, reducing bias towards majority classes. Instead of calculating the probability of an item belonging to a certain class, we calculate the probability of the item belonging to all the classes.
- Bernoulli NB is used for binary data, representing the presence or absence of words. Assumes features are binary indicators (0 or 1) of whether a word occurs in a document. Effective for document classification tasks that use binary features and spam detection.

Complement NB

```
print('ComplementNB model accuracy is',str('{:04.2f}'.format(accuracy_score*100))+'%')
print('-----')
print('Confusion Matrix:')
print(pd.DataFrame(confusion_matrix(y_test, predicted)))
print('-----')
print('Classification Report:')
print(classification_report(y_test, predicted))
```

ComplementNB model accuracy is 81.82%

Confusion Matrix:

	0	1
0	202	44
1	44	194

Multinomial NB

```
print('MultinomialNB model accuracy is',str('{:04.2f}'.format(accuracy_score*100))+'%')
print('-----')
print('Confusion Matrix:')
print(pd.DataFrame(confusion_matrix(y_test, predicted)))
print('-----')
print('Classification Report:')
print(classification_report(y_test, predicted))
```

MultinomialNB model accuracy is 82.02%

Confusion Matrix:

	0	1
0	202	44
1	43	195

Bernoulli NB

```
print('BernoulliNB model accuracy = ' + str('{:4.2f}'.format(accuracy_score_bnb*100))+ '%')
print('-----')
print('Confusion Matrix:')
print(pd.DataFrame(confusion_matrix(y_test, predicted)))
print('-----')
print('Classification Report:')
print(classification_report(y_test, predicted))
```

BernoulliNB model accuracy = 82.85%

Confusion Matrix:

	0	1
0	207	39
1	44	194

LDA

- The Latent Dirichlet Allocation (LDA) model identified distinct topics from the negative reviews after sentiment analysis.
- Each topic is represented by a collection of words that frequently appear together in the reviews.
- These words help to form a coherent theme or subject that the reviews focus on.
- For instance, some topics might revolve around specific issues such as "poor acting," "weak storyline," or "bad visual effects."
- These topics were automatically generated by the LDA model based on the patterns of word co-occurrence within the dataset.

Example

1. First, the reviews are tokenized into words, and a word count matrix is created.
2. LDA then analyzes the patterns in which these words appear together across the reviews. It tries to find groups (topics) of words that frequently co-occur.

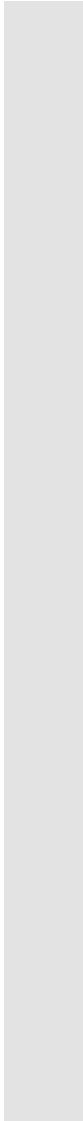
Let's say LDA identifies three topics from the negative reviews:

- **Topic 1:** Words related to **plot and story issues**.
- **Topic 2:** Words related to **acting and character development**.
- **Topic 3:** Words related to **technical aspects (e.g., special effects, direction)**

3. LDA will assign probabilities to each word, indicating how strongly each word is associated with each topic.



Now, LDA determines how much of each topic is present in each review:

- **Review 1:** Mostly about **Topic 2** (Acting & Characters) with a bit of **Topic 1** (Plot Issues).
 - **Review 2:** Mostly **Topic 3** (Technical Flaws) with some **Topic 1** (Plot Issues).
 - **Review 3:** Predominantly **Topic 2** (Acting & Characters).
 - **Review 4:** Strongly **Topic 3** (Technical Flaws).
 - **Review 5:** Primarily **Topic 1** (Plot Issues).
- 

Topic of Negative Reviews by LDA

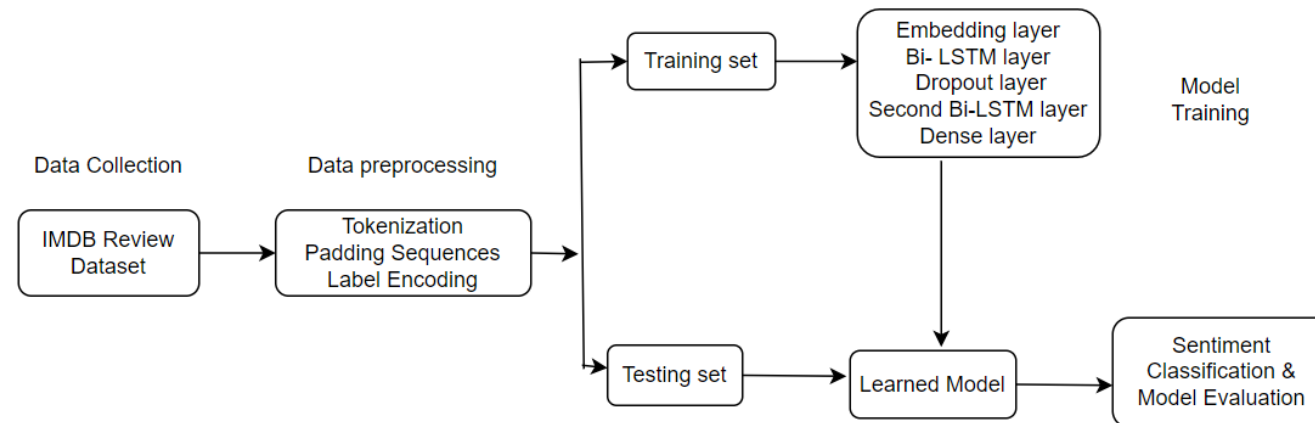
⇒

```
(0, '0.004*"jessica" + 0.003*"laurel" + 0.003*"stan" + 0.003*"mickey" + 0.003*"stephen" + 0.003*  
(1, '0.003*"cassavetes" + 0.003*"build" + 0.003*"touched" + 0.002*"damned" + 0.002*"daddy" + 0.0  
(2, '0.007*"plot" + 0.004*"scenes" + 0.004*"acting" + 0.004*"character" + 0.004*"watch" + 0.004*  
(3, '0.006*"funny" + 0.005*"original" + 0.003*"cast" + 0.003*"actors" + 0.003*"character" + 0.00  
(4, '0.005*"woman" + 0.004*"cannibal" + 0.004*"david" + 0.003*"young" + 0.002*"years" + 0.002*"a  
(5, '0.004*"back" + 0.003*"werewolf" + 0.003*"funny" + 0.003*"disney" + 0.003*"worst" + 0.002*"p  
(6, '0.003*"value" + 0.003*"expecting" + 0.002*"kevin" + 0.002*"truck" + 0.002*"williams" + 0.00  
(7, '0.007*"virus" + 0.005*"system" + 0.003*"drugs" + 0.003*"brainless" + 0.003*"frequently" + 0  
(8, '0.006*"game" + 0.004*"christian" + 0.004*"janeane" + 0.003*"standard" + 0.003*"chaplin" + 0  
(9, '0.008*"acting" + 0.007*"watch" + 0.007*"plot" + 0.006*"scene" + 0.006*"character" + 0.005*"  
LDA is not required for positive reviews.
```

Approach-2

Sentiment Analysis by
Bidirectional LSTM
followed by LSA for
Topic Modeling

- Bidirectional LSTM



Bidirectional LSTM Model

+ Code + Text

Connect T4 ▼

```
[ ] Max sequence length: 997
Average sequence length: 77.07674
Median sequence length: 57.0
Epoch 1/5
500/500 [=====] - 72s 133ms/step - loss: 0.3758 - accuracy: 0.8264 - val_loss: 0.3180 - val_accuracy: 0.8662
Epoch 2/5
500/500 [=====] - 29s 59ms/step - loss: 0.1935 - accuracy: 0.9285 - val_loss: 0.3553 - val_accuracy: 0.8590
Epoch 3/5
500/500 [=====] - 18s 36ms/step - loss: 0.1091 - accuracy: 0.9627 - val_loss: 0.4227 - val_accuracy: 0.8529
Epoch 4/5
500/500 [=====] - 15s 29ms/step - loss: 0.0611 - accuracy: 0.9804 - val_loss: 0.5701 - val_accuracy: 0.8525
Epoch 5/5
500/500 [=====] - 14s 28ms/step - loss: 0.0451 - accuracy: 0.9867 - val_loss: 0.6062 - val_accuracy: 0.8465
313/313 [=====] - 3s 7ms/step
Bidirectional LSTM Model Accuracy: 85.89%
Confusion Matrix:
      0      1
0  4174   787
1   624  4415
Classification Report:
              precision    recall  f1-score   support

     0       0.87         0.84         0.86         4961
     1       0.85         0.88         0.86         5039

    accuracy: 0.86         10000
```

LSA

- Latent Semantic Analysis (LSA) approaches topic modeling differently by capturing the underlying structure in the text data through singular value decomposition (SVD).
- The LSA model generates topics by identifying patterns in the relationships between terms in the dataset.
- The topics generated by LSA are generally more abstract and may capture broader themes.
- For example, the LSA model might generate topics that encompass general sentiments like "disappointment" or "lack of engagement," which can be attributed to various specific issues such as plot, acting, or pacing.

Example

- **Sample Negative Reviews:**

1. The acting was terrible, and the plot was confusing
2. I couldn't stand the love scenes, and the action was boring
3. This horror movie was not scary at all, just blood and no story
4. The series started off interesting, but the last few episodes were extremely boring
5. The characters were one-dimensional, and the dialogue was cringe-worthy

- Document-Term Matrix

Review	acting	terrible	plot	confusing	love	scenes	action	boring	horror	scar	Blood	Story	characters	dialogue
1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	1	1	1	1	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	1	1	1	1	0	0
4	0	0	0	0	0	0	0	1	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	1	1

- Applying Singular Value Decomposition (SVD)
- Dimensionality Reduction

- LSA uses Singular Value Decomposition (SVD) to reduce the dimensionality of the DTM. This means it breaks down the matrix into three smaller matrices (U , Σ , and V^T), capturing the relationships between terms and documents in a more compact form.
- **Matrix U :** Represents the document-topic relationships.
- **Matrix Σ :** Contains the singular values, which indicate the strength of each topic.
- **Matrix V^T :** Represents the term-topic relationships.
- This step identifies the most significant patterns in how terms co-occur across the documents.

- After SVD, you can choose to keep only the most significant components (topics) that capture the most important relationships in the data. For example, you might decide to keep only the top 2 or 3 topics if they capture the majority of the variance in the data.

Topic of Negative Reviews by LSA



Topics for Negative Reviews:

Topic 0: acting plot watch book scenes pretty life real character funny

Topic 1: series david gathering week episodes spanish hospital original televi

Topic 2: funny watch laugh find tv series extremely boring ive interesting

Topic 3: action love scenes jackie screen padrino pure masterpiece chan glad

Topic 4: book read horror blood ive bradbury reason saras witch acting

Topic 5: heat feel theme real series adventure song wonderful life audience

Topic 6: woman pretty find laugh book start dumb friends police big

Topic 7: japanese woman knotts horror makes star pretty dull roles kid

```
coherence_model_lsa = CoherenceModel(  
    model=None,  
    texts=texts,  
    dictionary=dictionary,  
    coherence='c_v',  
    topn=2,  
    topics=topic_terms  
)  
coherence_lsa = coherence_model_lsa.get_coherence()  
print(f'LSA Model Coherence Score: {coherence_lsa}')
```



LSA Model Coherence Score: 0.7883134196936762

Comparisons

Table 3.1: Results of Sentiment Analysis

Sentiment Analysis	Accuracy	Precision	Recall	F1Ratio
ComplementNB	0.8182	0.8211	0.8211	0.8228
MultinomialNB	0.8202	0.8211	0.8245	0.8211
BernoulliNB	0.8285	0.8415	0.8247	0.8330
BiDirectional LSTM	0.8589	0.8496	0.8751	0.8621

Table 3.2: Results of the Topic Modelling methods

Topic Modelling	Coherence
LDA	0.68
LSA	0.7883

References

- [1] Feng Li, Minlie Huang, and Xiaoyan Zhu. Sentiment analysis with global topics and local dependency. In Proceedings of AAAI, pages 1371–1376, 2010.
- [2] Chia-Hsiu Lin, Yifan He, Richard Everson, and Simon Ruger. Weakly supervised joint sentiment-topic detection from text. IEEE Transactions on Knowledge and Data Engineering, 24(6):1134–1145, 2012.
- [3] Tariq A. Rana, Y.-N. Cheah, and S. Letchmunan. Topic modeling in sentiment analysis: A systematic review. Journal of ICT Research and Applications, 10(1):76–93, 2016.
- [4] Qiaozhu Mei, Xue Ling, Michael Wondra, Hongfang Su, and ChengXiang Zhai. Topic-sentiment mixture: Modelling facets and opinions in weblogs. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 171–180, 2007.
- [5] Chao Xue, Wei Tang, Hong Xu, and Xian Hu. Double lda: A sentiment analysis model based on topic model. In Proceedings of the 2014 10th International Conference on Semantics, Knowledge and Grids, pages 49–56. IEEE Computer Society, 2014.
- [6] Daniel Jurafsky and James H. Martin. Speech and Language Processing. Draft of August 20, 2024, 2024. Copyright c 2024. All rights reserved.



THANK YOU